Machine Learning:
Modal Selection, Making ML Algorithms Work and ML Zoo

Ambedkar Dukkipati
Department of Computer Science and Automation
Indian Institute of Science, Bangalore

May 28, 2021

Agenda

Model Selection

Making ML Algorithms Work

Rewind

ML Zoo

Model Selection

Model Selection

- **Problem:** Given a set of models $\mathcal{M} = \{M_1, M_2, M_3.....M_\Delta\}$ choose the model that is expected to do the best on the test data

- Model selection can be of two types:
  - **Intra model selection:** Instances of same model with different complexities or hyperparameters or different sizes
  - **Inter model selection:** Different types of learning models

Model Selection

- **Problem:** Given a set of models $\mathcal{M} = \{M_1, M_2, M_3.....M_\Delta\}$ choose the model that is expected to do the best on the test data

- Model selection can be of two types:
  - **Intra model selection:** Instances of same model with different complexities or hyperparameters or different sizes
  - **Inter model selection:** Different types of learning models

Model Selection

- **Problem:** Given a set of models $\mathcal{M} = \{M_1, M_2, M_3.....M_\Delta\}$ choose the model that is expected to do the best on the test data

- Model selection can be of two types:
  - **Intra model selection:** Instances of same model with different complexities or hyperparameters or different sizes
  - **Inter model selection:** Different types of learning models

Model Selection (contd. . .)

- **Intra model selection**
  - K-Nearest Neighbours: Different choices of K
  - Polynomial Regression: Different degrees
  - Neural Networks: Number of layers
  - Decision Trees: Different number of leaves
  - Kernel Methods: Different choices of kernels
- **Inter model selection**
  - SVM, KNN, DT ?
- Model selection in unsupervised learning
  - Choosing the number of clusters (i.e number of components in Gaussian Mixture Models)

Model Selection (contd. . .)

- **Intra model selection**
  - K-Nearest Neighbours: Different choices of K
  - Polynomial Regression: Different degrees
  - Neural Networks: Number of layers
  - Decision Trees: Different number of leaves
  - Kernel Methods: Different choices of kernels
- **Inter model selection**
  - SVM, KNN, DT ?
- Model selection in unsupervised learning
  - Choosing the number of clusters (i.e number of components in Gaussian Mixture Models)

Model Selection (contd. . .)

- **Intra model selection**
  - K-Nearest Neighbours: Different choices of K
  - Polynomial Regression: Different degrees
  - Neural Networks: Number of layers
  - Decision Trees: Different number of leaves
  - Kernel Methods: Different choices of kernels
- **Inter model selection**
  - SVM, KNN, DT ?
- Model selection in unsupervised learning
  - Choosing the number of clusters (i.e number of components in Gaussian Mixture Models)

Model Selection (contd. . .)

- **Intra model selection**
  - K-Nearest Neighbours: Different choices of K
  - Polynomial Regression: Different degrees
  - Neural Networks: Number of layers
  - Decision Trees: Different number of leaves
  - Kernel Methods: Different choices of kernels
- **Inter model selection**
  - SVM, KNN, DT ?
- Model selection in unsupervised learning
  - Choosing the number of clusters (i.e number of components in Gaussian Mixture Models)

Model Selection (contd. . .)

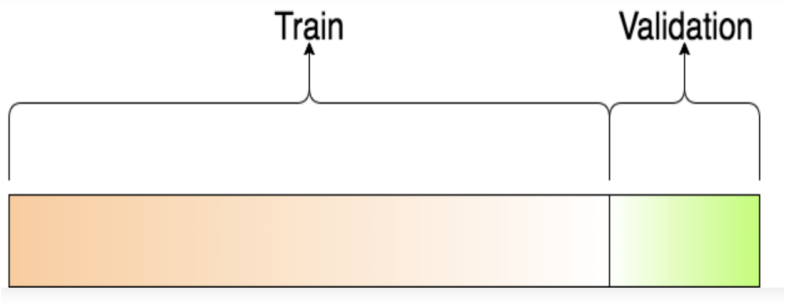
- **Intra model selection**
  - K-Nearest Neighbours: Different choices of K
  - Polynomial Regression: Different degrees
  - Neural Networks: Number of layers
  - Decision Trees: Different number of leaves
  - Kernel Methods: Different choices of kernels
- **Inter model selection**
  - SVM, KNN, DT ?
- Model selection in unsupervised learning
  - Choosing the number of clusters (i.e number of components in Gaussian Mixture Models)

Model Selection (contd. . .)

- **Intra model selection**
  - K-Nearest Neighbours: Different choices of K
  - Polynomial Regression: Different degrees
  - Neural Networks: Number of layers
  - Decision Trees: Different number of leaves
  - Kernel Methods: Different choices of kernels
- **Inter model selection**
  - SVM, KNN, DT ?
- Model selection in unsupervised learning
  - Choosing the number of clusters (i.e number of components in Gaussian Mixture Models)

Validation or Held Out Data

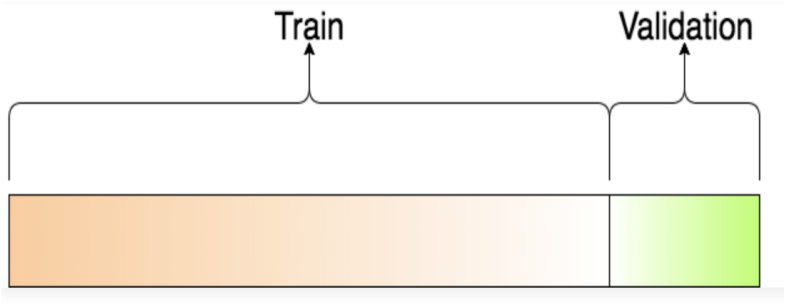
- **Held out data:** A fraction of the training data



- **Note:** Held out set is not the test data. Held out set is also known as validation set.

Validation or Held Out Data

- **Held out data:** A fraction of the training data



- **Note:** Held out set is not the test data. Held out set is also known as validation set.

Validation or Held Out Data (Cont. . .)

- **Cross Validation**
  - Train each model using remaining training data
  - Evaluate error on the held out set
  - Choose the model with the smallest error on held out set
- Issues
  - Wastage of training data
  - What if the split is not appropriate?

Validation or Held Out Data (Cont. . .)

- **Cross Validation**
  - Train each model using remaining training data
  - Evaluate error on the held out set
  - Choose the model with the smallest error on held out set
- Issues
  - Wastage of training data
  - What if the split is not appropriate?

Validation or Held Out Data (Cont. . .)

- **Cross Validation**
  - Train each model using remaining training data
  - Evaluate error on the held out set
  - Choose the model with the smallest error on held out set
- **Issues**
  - Wastage of training data
  - What if the split is not appropriate?

Validation or Held Out Data (Cont...)

- **Cross Validation**
  - Train each model using remaining training data
  - Evaluate error on the held out set
  - Choose the model with the smallest error on held out set
- **Issues**
  - Wastage of training data
  - What if the split is not appropriate?

K-fold Cross-Validation

- Create K equal sized partitions of the training data *i.e.*, each partition will have N/K examples
- Train using $K-1$ partitions and validate using the remaining partition
- Repeat this K times, each with different validation partitions
- Average K validation errors
- Choose the model that gives smallest average validation error

K-fold Cross-Validation

- Create K equal sized partitions of the training data *i.e.*, each partition will have N/K examples
- Train using $K-1$ partitions and validate using the remaining partition
- Repeat this K times, each with different validation partitions
- Average K validation errors
- Choose the model that gives smallest average validation error

K-fold Cross-Validation

- Create K equal sized partitions of the training data *i.e.*, each partition will have N/K examples
- Train using $K-1$ partitions and validate using the remaining partition
- Repeat this K times, each with different validation partitions
- Average K validation errors
- Choose the model that gives smallest average validation error

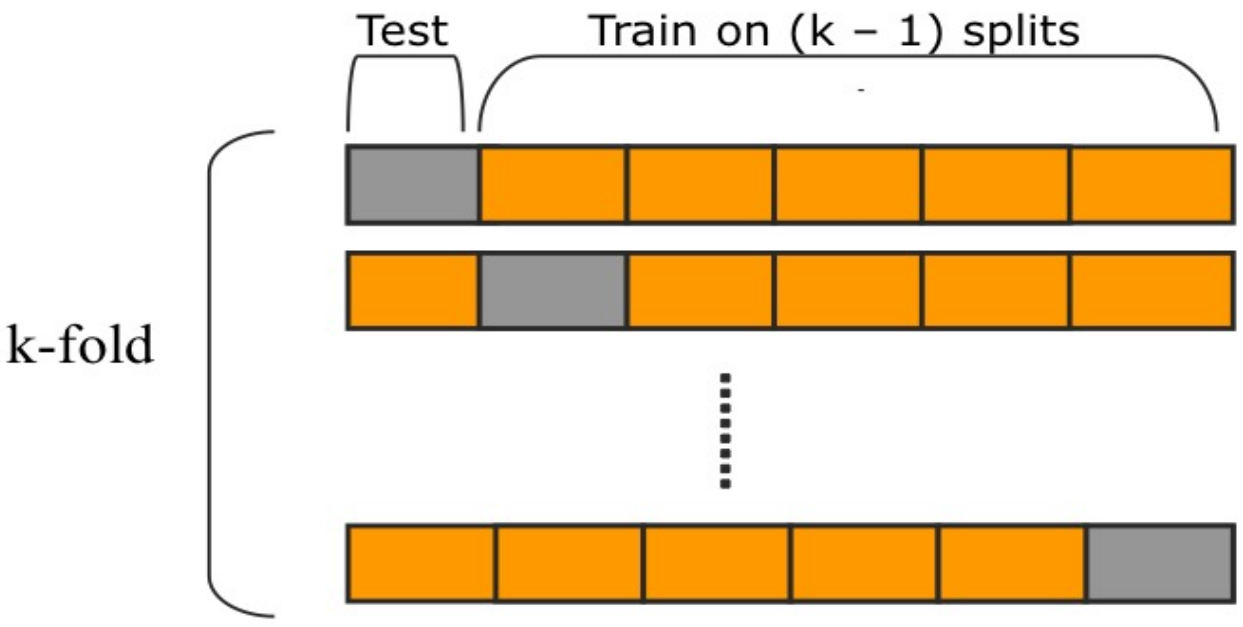
K-fold Cross-Validation

- Create K equal sized partitions of the training data *i.e.*, each partition will have N/K examples
- Train using $K-1$ partitions and validate using the remaining partition
- Repeat this K times, each with different validation partitions
- Average K validation errors
- Choose the model that gives smallest average validation error

K-fold Cross-Validation

- Create K equal sized partitions of the training data *i.e.*, each partition will have N/K examples
- Train using $K-1$ partitions and validate using the remaining partition
- Repeat this K times, each with different validation partitions
- Average K validation errors
- Choose the model that gives smallest average validation error

K-fold Cross-Validation (Cont. . .)



Leave-One-Out Cross-Validation

- Now validation set has just one example
- Train using $N-1$ examples and validate using the remaining example
- Average the N validation errors and choose the model that gives smallest average validation error
- **Note:** Can be very expensive for large N
- Works well for neighbourhood based methods (since training time is less)

Leave-One-Out Cross-Validation

- Now validation set has just one example
- Train using $N - 1$ examples and validate using the remaining example
- Average the N validation errors and choose the model that gives smallest average validation error
- **Note:** Can be very expensive for large N
- Works well for neighbourhood based methods (since training time is less)

Leave-One-Out Cross-Validation

- Now validation set has just one example
- Train using $N - 1$ examples and validate using the remaining example
- Average the N validation errors and choose the model that gives smallest average validation error
- **Note:** Can be very expensive for large N
- Works well for neighbourhood based methods (since training time is less)

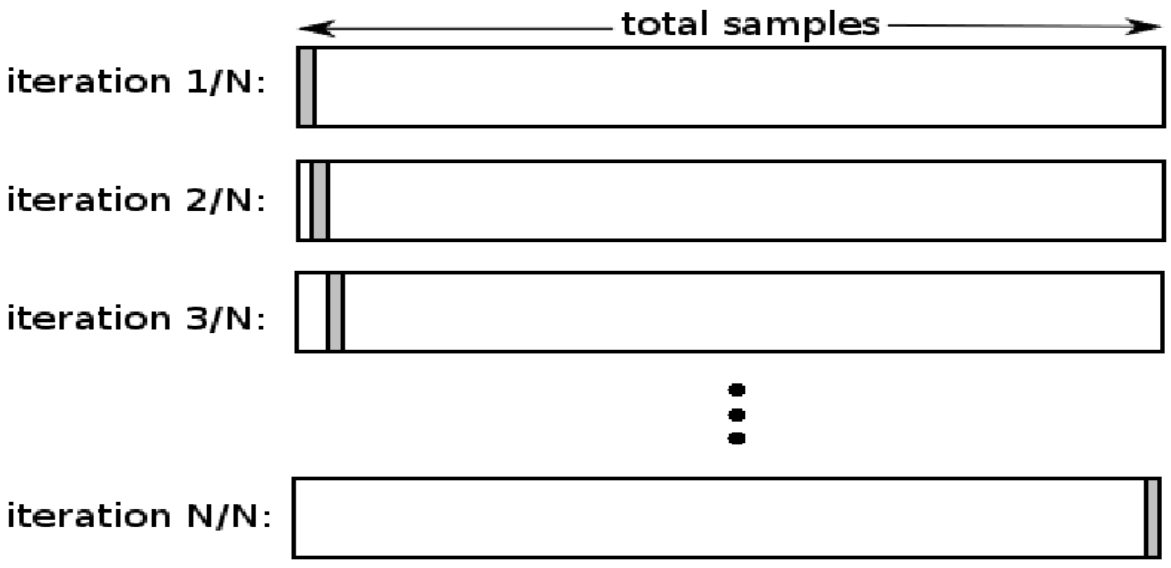
Leave-One-Out Cross-Validation

- Now validation set has just one example
- Train using $N - 1$ examples and validate using the remaining example
- Average the N validation errors and choose the model that gives smallest average validation error
- **Note:** Can be very expensive for large N
- Works well for neighbourhood based methods (since training time is less)

Leave-One-Out Cross-Validation

- Now validation set has just one example
- Train using $N - 1$ examples and validate using the remaining example
- Average the N validation errors and choose the model that gives smallest average validation error
- **Note:** Can be very expensive for large N
- Works well for neighbourhood based methods (since training time is less)

Leave-One-Out Cross-Validation (Cont. . .)



Random Sampling based Cross-Validation

- Subsample a fixed fraction αN $(0 < \alpha < 1)$ as examples of validation set
- Train using rest of the examples and calculate the validation error
- Repeat K times, each with a different, randomly chosen validation set
- Average the K validation errors and choose the model that gives smallest average validation error

Random Sampling based Cross-Validation

- Subsample a fixed fraction αN $(0 < \alpha < 1)$ as examples of validation set
- Train using rest of the examples and calculate the validation error
- Repeat K times, each with a different, randomly chosen validation set
- Average the K validation errors and choose the model that gives smallest average validation error

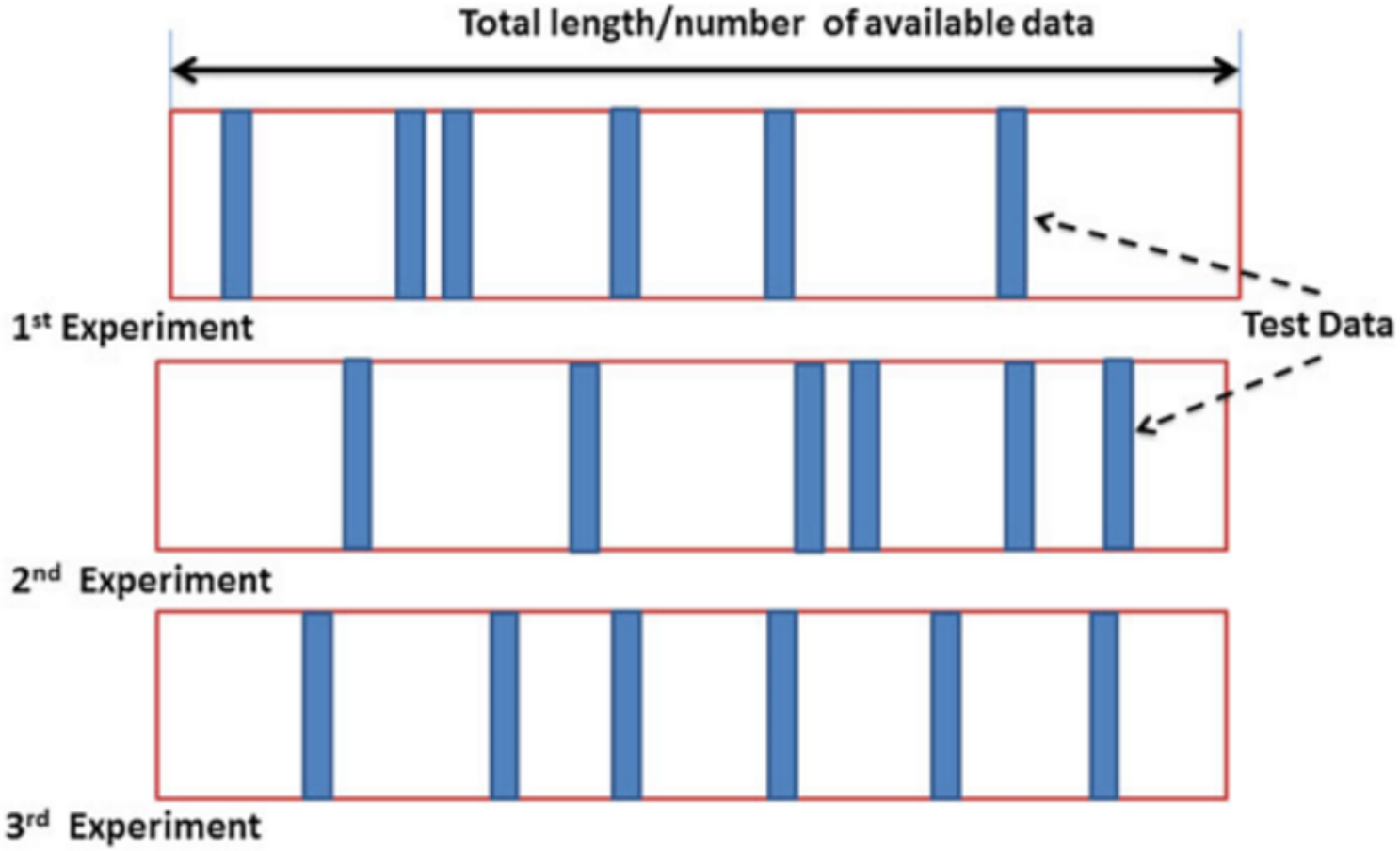
Random Sampling based Cross-Validation

- Subsample a fixed fraction αN $(0 < \alpha < 1)$ as examples of validation set
- Train using rest of the examples and calculate the validation error
- Repeat K times, each with a different, randomly chosen validation set
- Average the K validation errors and choose the model that gives smallest average validation error

Random Sampling based Cross-Validation

- Subsample a fixed fraction αN $(0 < \alpha < 1)$ as examples of validation set
- Train using rest of the examples and calculate the validation error
- Repeat K times, each with a different, randomly chosen validation set
- Average the K validation errors and choose the model that gives smallest average validation error

Random Sampling based Cross-Validation (Cont. . .)



Bootstrap

- **Main Idea:** Given N examples, sample N elements with replacement i.e samples can be repeated
- Use the N examples as training data
- Use the set of examples not selected as the validation data
- For large N, training data will have 63% unique examples

$$\because \text{Fraction of examples not picked}$$
$$(1 - 1/N)^N \approx e^{-1} \approx 0.368$$

- $Error = (0.632) * err_{test\ examples} + (0.368) * err_{train\ examples}$

Bootstrap

- **Main Idea:** Given N examples, sample N elements with replacement i.e samples can be repeated
- Use the N examples as training data
- Use the set of examples not selected as the validation data
- For large N, training data will have 63% unique examples

$$\because \text{Fraction of examples not picked}$$
$$(1 - 1/N)^N \approx e^{-1} \approx 0.368$$

- $Error = (0.632) * err_{test\ examples} + (0.368) * err_{train\ examples}$

Bootstrap

- **Main Idea:** Given N examples, sample N elements with replacement i.e samples can be repeated
- Use the N examples as training data
- Use the set of examples not selected as the validation data
- For large N, training data will have 63% unique examples

$$\because \text{Fraction of examples not picked}$$
$$(1 - 1/N)^N \approx e^{-1} \approx 0.368$$

- $Error = (0.632) * err_{test\ examples} + (0.368) * err_{train\ examples}$

Bootstrap

- **Main Idea:** Given N examples, sample N elements with replacement i.e samples can be repeated
- Use the N examples as training data
- Use the set of examples not selected as the validation data
- For large N, training data will have 63% unique examples

$\because$ Fraction of examples not picked

$$(1 - 1/N)^N \approx e^{-1} \approx 0.368$$

- $Error = (0.632) * err_{test\ examples} + (0.368) * err_{train\ examples}$

Bootstrap

- **Main Idea:** Given N examples, sample N elements with replacement i.e samples can be repeated
- Use the N examples as training data
- Use the set of examples not selected as the validation data
- For large N, training data will have 63% unique examples

$$\because \text{Fraction of examples not picked}$$
$$(1 - 1/N)^N \approx e^{-1} \approx 0.368$$

- $Error = (0.632) * err_{test\ examples} + (0.368) * err_{train\ examples}$

On 63 percent . . .

- Given N examples, sample N elements with replacement.
- **FACT:**
  - For large N training data consists of about only 63% unique examples.
  - Probability that an example not picked $= 1 - \frac{1}{N}$
  - Fractions of examples not picked $(1 - \frac{1}{N})^N = \frac{1}{e} \approx 0.368$.
- **FACT:**
  - $\lim_{N \to \infty} (1 + \frac{1}{N})^N = e$
  - Let t be any number in an interval $[1, 1 + \frac{1}{N}]$. Then $\frac{1}{1+\frac{1}{N}} \leq \frac{1}{t} \leq 1$

On 63 percent . . .

- Given N examples, sample N elements with replacement.
- **FACT:**
  - For large N training data consists of about only 63% unique examples.
  - Probability that an example not picked $= 1 - \frac{1}{N}$
  - Fractions of examples not picked $(1 - \frac{1}{N})^N = \frac{1}{e} \approx 0.368$.
- **FACT:**
  - $\lim_{N \to \infty} (1 + \frac{1}{N})^N = e$
  - Let t be any number in an interval $[1, 1 + \frac{1}{N}]$. Then $\frac{1}{1+\frac{1}{N}} \leq \frac{1}{t} \leq 1$

On 63 percent . . .

- Given N examples, sample N elements with replacement.
- **FACT:**
  - For large N training data consists of about only 63% unique examples.
  - Probability that an example not picked $= 1 - \frac{1}{N}$
  - Fractions of examples not picked $(1 - \frac{1}{N})^N = \frac{1}{e} \approx 0.368$.
- **FACT:**
  - $\lim_{N \to \infty} (1 + \frac{1}{N})^N = e$
  - Let t be any number in an interval $[1, 1 + \frac{1}{N}]$. Then $\frac{1}{1+\frac{1}{N}} \leq \frac{1}{t} \leq 1$

On 63 percent . . .

- Given N examples, sample N elements with replacement.
- **FACT:**
  - For large N training data consists of about only 63% unique examples.
  - Probability that an example not picked $= 1 - \frac{1}{N}$
  - Fractions of examples not picked $(1 - \frac{1}{N})^N = \frac{1}{e} \approx 0.368$.
- **FACT:**
  - $\lim_{N \to \infty}(1 + \frac{1}{N})^N = e$
  - Let t be any number in an interval $[1, 1 + \frac{1}{N}]$. Then $\frac{1}{1+\frac{1}{N}} \leq \frac{1}{t} \leq 1$

On 63 percent . . .

- Given N examples, sample N elements with replacement.
- **FACT:**
  - For large N training data consists of about only 63% unique examples.
  - Probability that an example not picked $= 1 - \frac{1}{N}$
  - Fractions of examples not picked $(1 - \frac{1}{N})^N = \frac{1}{e} \approx 0.368$.
- **FACT:**
  - $\lim_{N \to \infty}(1 + \frac{1}{N})^N = e$
  - Let t be any number in an interval $[1, 1 + \frac{1}{N}]$. Then $\frac{1}{1+\frac{1}{N}} \leq \frac{1}{t} \leq 1$

On 63 percent (cont. . .)

$$\implies \int_1^{1+\frac{1}{N}} \frac{1}{1+\frac{1}{N}} dt \leq \int_1^{1+\frac{1}{N}} \frac{1}{t} dt \leq \int_1^{1+\frac{1}{N}} dt$$

$$\implies \frac{1}{1+N} \leq \ln(1+\frac{1}{N}) \leq \frac{1}{N}$$

$$\implies e^{1+\frac{1}{N}} \leq 1+\frac{1}{N} \leq e^{\frac{1}{N}}$$

$$\implies e \leq (1+\frac{1}{N})^{N+1} \text{ and } (1+\frac{1}{N})^{N} \leq e$$

Divide right inequality with $(1+\frac{1}{N})$ which gives,

$$(\frac{e}{1+\frac{1}{N}}) \leq (1+\frac{1}{N})^{N} \leq e$$

As $N \to \infty, (\frac{e}{1+\frac{1}{N}}) \to e$, Hence proved

Information Theoretic Methods

- **Notations:**
  - - K : number of model parameters
  - - L : maximum likelihood of observed data under the model
- **Occam's Razor:** Among all possible explanations pick up the most simplest one
- **Akaike Information Criteria (AIC)**

$$AIC = 2K - 2\log(L)$$

- **Bayesian Information Criteria (BIC)**

$$BIC = K\log(N) - 2\log(L)$$

- AIC and BIC are applicable for probabilistic models
- AIC and BIC penalize the model complexity
- **Minimum Description Length:** Beyond this course

Information Theoretic Methods

- **Notations:**
  - - K : number of model parameters
  - - L : maximum likelihood of observed data under the model
- **Occam's Razor:** Among all possible explanations pick up the most simplest one
- **Akaike Information Criteria (AIC)**

$$AIC = 2K - 2\log(L)$$

- **Bayesian Information Criteria (BIC)**

$$BIC = K\log(N) - 2\log(L)$$

- AIC and BIC are applicable for probabilistic models
- AIC and BIC penalize the model complexity
- **Minimum Description Length:** Beyond this course

Information Theoretic Methods

- **Notations:**
  - - K : number of model parameters
  - - L : maximum likelihood of observed data under the model
- **Occam's Razor:** Among all possible explanations pick up the most simplest one
- **Akaike Information Criteria (AIC)**

$$AIC = 2K - 2\log(L)$$

- **Bayesian Information Criteria (BIC)**

$$BIC = K\log(N) - 2\log(L)$$

- AIC and BIC are applicable for probabilistic models
- AIC and BIC penalize the model complexity
- **Minimum Description Length:** Beyond this course

Information Theoretic Methods

- **Notations:**
  - - K : number of model parameters
  - - L : maximum likelihood of observed data under the model
- **Occam's Razor:** Among all possible explanations pick up the most simplest one
- **Akaike Information Criteria (AIC)**

$$AIC = 2K - 2\log(L)$$

- **Bayesian Information Criteria (BIC)**

$$BIC = K\log(N) - 2\log(L)$$

- AIC and BIC are applicable for probabilistic models
- AIC and BIC penalize the model complexity
- **Minimum Description Length:** Beyond this course

Information Theoretic Methods

- **Notations:**
  - - K : number of model parameters
  - - L : maximum likelihood of observed data under the model
- **Occam's Razor:** Among all possible explanations pick up the most simplest one
- **Akaike Information Criteria (AIC)**

$$AIC = 2K - 2\log(L)$$

- **Bayesian Information Criteria (BIC)**

$$BIC = K\log(N) - 2\log(L)$$

- AIC and BIC are applicable for probabilistic models
- AIC and BIC penalize the model complexity
- **Minimum Description Length:** Beyond this course

Information Theoretic Methods

- **Notations:**
  - - K : number of model parameters
  - - L : maximum likelihood of observed data under the model
- **Occam's Razor:** Among all possible explanations pick up the most simplest one
- **Akaike Information Criteria (AIC)**

$$AIC = 2K - 2\log(L)$$

- **Bayesian Information Criteria (BIC)**

$$BIC = K\log(N) - 2\log(L)$$

- AIC and BIC are applicable for probabilistic models
- AIC and BIC penalize the model complexity
- **Minimum Description Length:** Beyond this course

Information Theoretic Methods

- **Notations:**
  - - K : number of model parameters
  - - L : maximum likelihood of observed data under the model
- **Occam's Razor:** Among all possible explanations pick up the most simplest one
- **Akaike Information Criteria (AIC)**

$$AIC = 2K - 2\log(L)$$

- **Bayesian Information Criteria (BIC)**

$$BIC = K\log(N) - 2\log(L)$$

- AIC and BIC are applicable for probabilistic models
- AIC and BIC penalize the model complexity
- **Minimum Description Length:** Beyond this course

Information Theoretic Methods(contd. . .)

- Here model complexity is measured by the number of model parameters
- BIC penalizes the number of parameters more than AIC
- **Occam's Razor:** Model with the lowest AIC or BIC is chosen
- AIC and BIC can be used in unsupervised learning

Information Theoretic Methods(contd. . .)

- Here model complexity is measured by the number of model parameters
- BIC penalizes the number of parameters more than AIC
- **Occam's Razor:** Model with the lowest AIC or BIC is chosen
- AIC and BIC can be used in unsupervised learning

Information Theoretic Methods(contd. . .)

- Here model complexity is measured by the number of model parameters
- BIC penalizes the number of parameters more than AIC
- **Occam's Razor:** Model with the lowest AIC or BIC is chosen
- AIC and BIC can be used in unsupervised learning

Information Theoretic Methods(contd. . .)

- Here model complexity is measured by the number of model parameters
- BIC penalizes the number of parameters more than AIC
- **Occam's Razor:** Model with the lowest AIC or BIC is chosen
- AIC and BIC can be used in unsupervised learning

Making ML Algorithms Work

On Making Learning Algorithms Work

- ML algorithms are data driven and not procedural.
  - It is reasonably difficult to make them work even if we are sure that our implementation is correct.
- What should one do if the model is not working "very well" *i.e.*, not getting acceptable levels of test accuracy.
  - **Note:** When training error is too high and the test accuracy is very less there is something wrong.
  - Generalization capacity is one of the most important aspect to look forward to in a model.

On Making Learning Algorithms Work

- ML algorithms are data driven and not procedural.
  - It is reasonably difficult to make them work even if we are sure that our implementation is correct.
- What should one do if the model is not working "very well" *i.e.*, not getting acceptable levels of test accuracy.
  - **Note:** When training error is too high and the test accuracy is very less there is something wrong.
  - Generalization capacity is one of the most important aspect to look forward to in a model.

On Making Learning Algorithms Work

- ML algorithms are data driven and not procedural.
  - It is reasonably difficult to make them work even if we are sure that our implementation is correct.
- What should one do if the model is not working "very well" *i.e.*, not getting acceptable levels of test accuracy.
  - **Note:** When training error is too high and the test accuracy is very less there is something wrong.
  - Generalization capacity is one of the most important aspect to look forward to in a model.

On Making Learning Algorithms Work

- ML algorithms are data driven and not procedural.
  - It is reasonably difficult to make them work even if we are sure that our implementation is correct.
- What should one do if the model is not working "very well" *i.e.*, not getting acceptable levels of test accuracy.
  - **Note:** When training error is too high and the test accuracy is very less there is something wrong.
  - Generalization capacity is one of the most important aspect to look forward to in a model.

On Making Learning Algorithms Work

- ML algorithms are data driven and not procedural.
  - It is reasonably difficult to make them work even if we are sure that our implementation is correct.
- What should one do if the model is not working "very well" *i.e.*, not getting acceptable levels of test accuracy.
  - **Note:** When training error is too high and the test accuracy is very less there is something wrong.
  - Generalization capacity is one of the most important aspect to look forward to in a model.

On Making Learning Algorithm Work (contd. . .)

- So what can we do?
  - Use more training examples to train a model.
  - Use a small number of features.
  - Introduce new features in the case of hand picked features.
  - Tune hyper-parameters like regularization parameters.
  - Optimize for more number of iterations.
  - Change the optimization algorithm (GD to SGD or Newton).
  - Give up and move to some other model.

On Making Learning Algorithm Work (contd. . .)

- So what can we do?
  - Use more training examples to train a model.
  - Use a small number of features.
  - Introduce new features in the case of hand picked features.
  - Tune hyper-parameters like regularization parameters.
  - Optimize for more number of iterations.
  - Change the optimization algorithm (GD to SGD or Newton).
  - Give up and move to some other model.

On Making Learning Algorithm Work (contd. . .)

- So what can we do?
  - Use more training examples to train a model.
  - Use a small number of features.
  - Introduce new features in the case of hand picked features.
  - Tune hyper-parameters like regularization parameters.
  - Optimize for more number of iterations.
  - Change the optimization algorithm (GD to SGD or Newton).
  - Give up and move to some other model.

On Making Learning Algorithm Work (contd. . .)

- So what can we do?
  - Use more training examples to train a model.
  - Use a small number of features.
  - Introduce new features in the case of hand picked features.
  - Tune hyper-parameters like regularization parameters.
  - Optimize for more number of iterations.
  - Change the optimization algorithm (GD to SGD or Newton).
  - Give up and move to some other model.

On Making Learning Algorithm Work (contd. . .)

- So what can we do?
  - Use more training examples to train a model.
  - Use a small number of features.
  - Introduce new features in the case of hand picked features.
  - Tune hyper-parameters like regularization parameters.
  - Optimize for more number of iterations.
  - Change the optimization algorithm (GD to SGD or Newton).
  - Give up and move to some other model.

On Making Learning Algorithm Work (contd. . .)

- So what can we do?
  - Use more training examples to train a model.
  - Use a small number of features.
  - Introduce new features in the case of hand picked features.
  - Tune hyper-parameters like regularization parameters.
  - Optimize for more number of iterations.
  - Change the optimization algorithm (GD to SGD or Newton).
  - Give up and move to some other model.

On Making Learning Algorithm Work (contd. . .)

- So what can we do?
  - Use more training examples to train a model.
  - Use a small number of features.
  - Introduce new features in the case of hand picked features.
  - Tune hyper-parameters like regularization parameters.
  - Optimize for more number of iterations.
  - Change the optimization algorithm (GD to SGD or Newton).
  - Give up and move to some other model.

On Making Learning Algorithm Work (contd. . .)

- So what can we do?
  - Use more training examples to train a model.
  - Use a small number of features.
  - Introduce new features in the case of hand picked features.
  - Tune hyper-parameters like regularization parameters.
  - Optimize for more number of iterations.
  - Change the optimization algorithm (GD to SGD or Newton).
  - Give up and move to some other model.

Bias and Variance

- **Setting:** Assume a supervised learning procedure.
  - Assume that model is

$$y = f(x) + \epsilon$$

  - Given some training data $\hat{f}$ denotes the estimate of f.
  - Assume that

$$\epsilon \sim \mathbb{N}(0, \sigma^2)$$

  - We have

$$\mathrm{E}[y|x] = f(x)$$

Bias and Variance

- **Setting:** Assume a supervised learning procedure.
  - Assume that model is

$$y = f(x) + \epsilon$$

  - Given some training data $\hat{f}$ denotes the estimate of f.
  - Assume that

$$\epsilon \sim \mathbb{N}(0, \sigma^2)$$

  - We have

$$\mathrm{E}[y|x] = f(x)$$

Bias and Variance

- **Setting:** Assume a supervised learning procedure.
  - Assume that model is

$$y = f(x) + \epsilon$$

  - Given some training data $\hat{f}$ denotes the estimate of f.
  - Assume that

$$\epsilon \sim \mathbb{N}(0, \sigma^2)$$

  - We have

$$\mathrm{E}[y|x] = f(x)$$

Bias and Variance

- **Setting:** Assume a supervised learning procedure.
  - Assume that model is

$$y = f(x) + \epsilon$$

  - Given some training data $\hat{f}$ denotes the estimate of f.
  - Assume that

$$\epsilon \sim \mathbb{N}(0, \sigma^2)$$

  - We have

$$\mathrm{E}[y|x] = f(x)$$

Bias and Variance

- **Setting:** Assume a supervised learning procedure.
  - Assume that model is

$$y = f(x) + \epsilon$$

  - Given some training data $\hat{f}$ denotes the estimate of f.
  - Assume that

$$\epsilon \sim \mathbb{N}(0, \sigma^2)$$

  - We have

$$\mathrm{E}[y|x] = f(x)$$

Bias and Variance (contd. . .)

- A simple calculation, conditioning on X has not been explicitly mentioned to avoid clutter.

$$
\begin{aligned}
\mathsf{E}[(y-\hat{f})^2] &= \mathsf{E}[y^2+\hat{f}^2-2y\hat{f}] \\
&= \mathsf{E}[y^2]+\mathsf{E}[\hat{f}^2]-\mathsf{E}[2y\hat{f}] \\
&= \mathsf{Var}[y]+\mathsf{E}[y]^2+\mathsf{Var}[\hat{f}]+\mathsf{E}[\hat{f}]^2-2f\mathsf{Var}[\hat{f}] \\
&= \mathsf{Var}[y]+\mathsf{Var}[\hat{f}]+(f-\mathsf{E}[\hat{f}])^2 \\
&= \mathsf{Var}[y]+\mathsf{Var}[\hat{f}]+\mathsf{E}[f-\hat{f}]^2 \\
&= \sigma^2+\mathsf{Var}[\hat{f}]+\mathsf{Bias}[\hat{f}]^2 \\
&= \sigma^2+\mathsf{Var}[\hat{f}(x)]+\mathsf{Bias}[\hat{f}(x)]^2
\end{aligned}
\tag{1}
$$

$\mathsf{Bias}[\hat{f}(x)] = \mathsf{E}[\hat{f}(x)-f(x)]$: Error due to wrong model.
$\mathsf{Var}[\hat{f}(x)] = \mathsf{E}[\hat{f}(x)^2]-\mathsf{E}[\hat{f}(x)]^2$: Learner's sensitivity to choice of training set.

Bias and Variance Trade-off

- **FACT:**
  - Simple model have high bias and small variance.
  - Complex model have small bias and high variance.

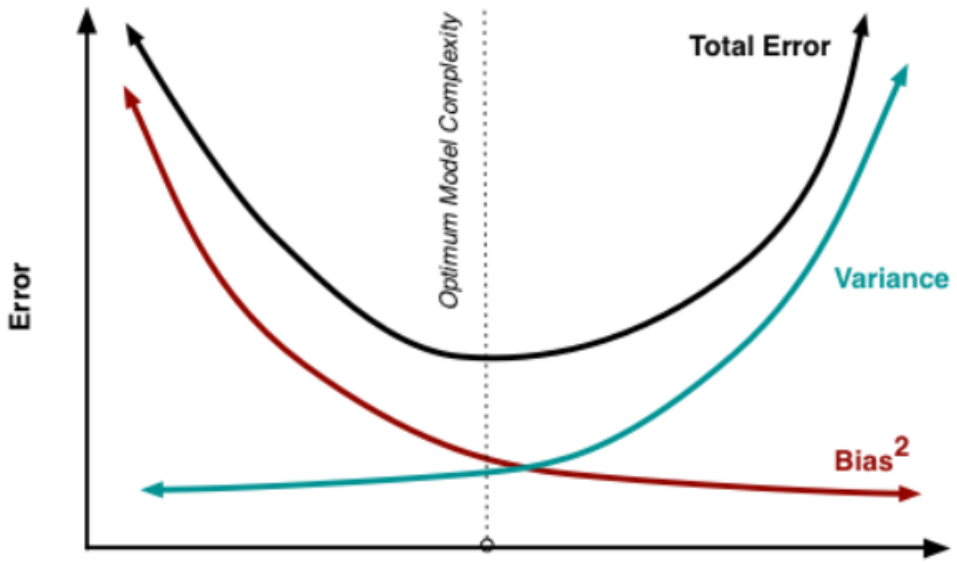


Figure 1: Model Complexity [1]

[1] Image Source: Scott Fortmann-Roe, Latysheva and Ravarani

Bias and Variance Trade-off

- **FACT:**
  - Simple model have high bias and small variance.
  - Complex model have small bias and high variance.

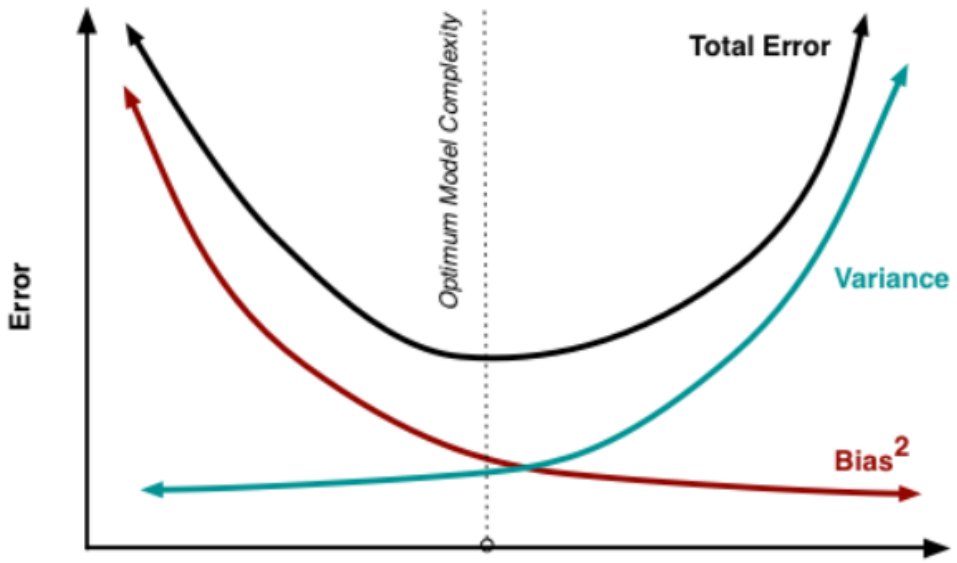


Figure 1: Model Complexity [1]

[1] Image Source: Scott Fortmann-Roe, Latysheva and Ravarani

Bias and Variance Trade-off

- **FACT:**
  - Simple model have high bias and small variance.
  - Complex model have small bias and high variance.

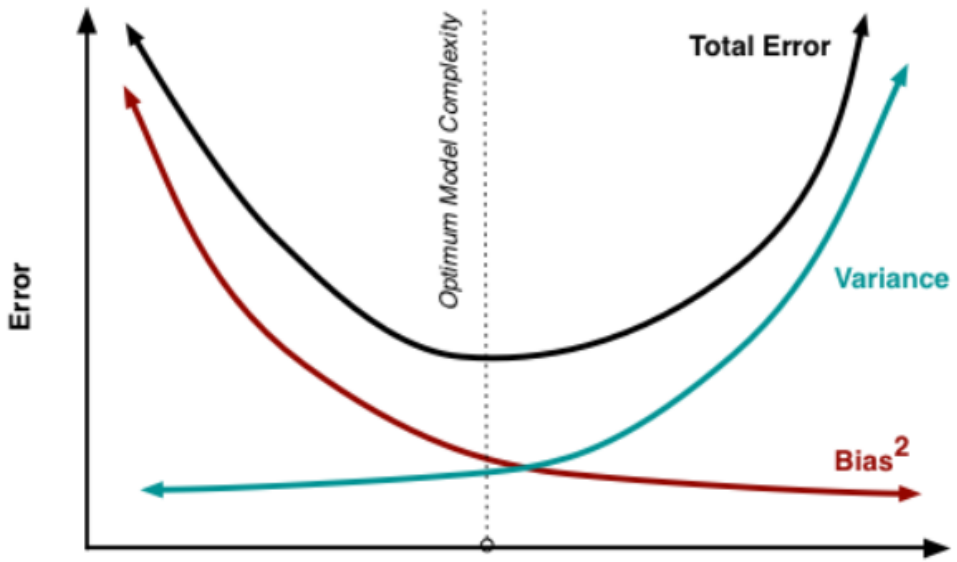


Figure 1: Model Complexity [1]

[1] Image Source: Scott Fortmann-Roe, Latysheva and Ravarani

Bias and Variance Trade-off (contd. . .)

	Bias	**Variance**	**Complexity**	**Flexibility**	**Generalizabil**
Underfitting:Very simple model	High	Low	Low	Low	High[2]
Overfitting:Very complex model	Low	High	High	High	Low

- If we try to reduce bias by increasing the model complexity, the variance will increase and vice versa.

[2] Assuming a reasonable model

Bias and Variance Trade-off (contd. . .)

	Bias	**Variance**	**Complexity**	**Flexibility**	**Generalizabil**
Underfitting:Very simple model	High	Low	Low	Low	High[2]
Overfitting:Very complex model	Low	High	High	High	Low

- If we try to reduce bias by increasing the model complexity, the variance will increase and vice versa.

[2] Assuming a reasonable model

Bias and Variance Trade-off (contd. . .)

- Low accuracy on test data can be due to either
  - High Bias (Under-fitting)
  - High Variance (Over-fitting)
- Training error and test error can give the diagnosis.
- High Bias: Both training and test error are large.
- High Variance: Small training error, large test error.

Bias and Variance Trade-off (contd. . .)

- Low accuracy on test data can be due to either
  - High Bias (Under-fitting)
  - High Variance (Over-fitting)
- Training error and test error can give the diagnosis.
- High Bias: Both training and test error are large.
- High Variance: Small training error, large test error.

Bias and Variance Trade-off (contd. . .)

- Low accuracy on test data can be due to either
  - High Bias (Under-fitting)
  - High Variance (Over-fitting)
- Training error and test error can give the diagnosis.
- High Bias: Both training and test error are large.
- High Variance: Small training error, large test error.

Bias and Variance Trade-off (contd. . .)

- Low accuracy on test data can be due to either
  - High Bias (Under-fitting)
  - High Variance (Over-fitting)
- Training error and test error can give the diagnosis.
- High Bias: Both training and test error are large.
- High Variance: Small training error, large test error.

Bias and Variance Trade-off (contd. . .)

- Low accuracy on test data can be due to either
  - High Bias (Under-fitting)
  - High Variance (Over-fitting)
- Training error and test error can give the diagnosis.
- High Bias: Both training and test error are large.
- High Variance: Small training error, large test error.

Bias and Variance Trade-off (contd. . .)

- Low accuracy on test data can be due to either
  - High Bias (Under-fitting)
  - High Variance (Over-fitting)
- Training error and test error can give the diagnosis.
- High Bias: Both training and test error are large.
- High Variance: Small training error, large test error.

Bias and Variance Trade-off (contd. . .)

- If Bias is high
  - Adding more training examples will not usually bring the bias down.
  - Try making model more expressive, *e.g.*, adding more features or using more complicated model.
- If Variance is high
  - Using more training data can bring down the variance.
  - Other strategy is to make model simpler.

Bias and Variance Trade-off (contd. . .)

- If Bias is high
  - Adding more training examples will not usually bring the bias down.
  - Try making model more expressive, *e.g.*, adding more features or using more complicated model.
- If Variance is high
  - Using more training data can bring down the variance.
  - Other strategy is to make model simpler.

Bias and Variance Trade-off (contd. . .)

- If Bias is high
  - Adding more training examples will not usually bring the bias down.
  - Try making model more expressive, *e.g.*, adding more features or using more complicated model.
- If Variance is high
  - Using more training data can bring down the variance.
  - Other strategy is to make model simpler.

Bias and Variance Trade-off (contd. . .)

- If Bias is high
  - Adding more training examples will not usually bring the bias down.
  - Try making model more expressive, *e.g.*, adding more features or using more complicated model.
- If Variance is high
  - Using more training data can bring down the variance.
  - Other strategy is to make model simpler.

Bias and Variance Trade-off (contd. . .)

- If Bias is high
  - Adding more training examples will not usually bring the bias down.
  - Try making model more expressive, *e.g.*, adding more features or using more complicated model.
- If Variance is high
  - Using more training data can bring down the variance.
  - Other strategy is to make model simpler.

Bias and Variance Trade-off (contd. . .)

- If Bias is high
  - Adding more training examples will not usually bring the bias down.
  - Try making model more expressive, *e.g.*, adding more features or using more complicated model.
- If Variance is high
  - Using more training data can bring down the variance.
  - Other strategy is to make model simpler.

Bias and Variance Trade-off (contd. . .)

- If Bias is high
  - Adding more training examples will not usually bring the bias down.
  - Try making model more expressive, *e.g.*, adding more features or using more complicated model.
- If Variance is high
  - Using more training data can bring down the variance.
  - Other strategy is to make model simpler.

Rewind

What we have learned so far

- Machine Learning Workflow
  - Data preprocessing
  - Feature extraction
  - Dimensionality Reduction
  - Training
  - Validation
  - Testing

What we have learned so far (Contd. . .)

- Distance based classification
- Bayes decision theory and Bayes classifier
- Supervised learning and some foundations
- Linear Regression and Logistic Regression
- Maximum Likelihood and Maximum Apriori estimates
- Overfitting and Regularization

What we have learned so far (Contd. . .)

- Distance based classification
- Bayes decision theory and Bayes classifier
- Supervised learning and some foundations
- Linear Regression and Logistic Regression
- Maximum Likelihood and Maximum Apriori estimates
- Overfitting and Regularization

What we have learned so far (Contd. . .)

- Distance based classification
- Bayes decision theory and Bayes classifier
- Supervised learning and some foundations
- Linear Regression and Logistic Regression
- Maximum Likelihood and Maximum Apriori estimates
- Overfitting and Regularization

What we have learned so far (Contd. . .)

- Distance based classification
- Bayes decision theory and Bayes classifier
- Supervised learning and some foundations
- Linear Regression and Logistic Regression
- Maximum Likelihood and Maximum Apriori estimates
- Overfitting and Regularization

What we have learned so far (Contd. . .)

- Distance based classification
- Bayes decision theory and Bayes classifier
- Supervised learning and some foundations
- Linear Regression and Logistic Regression
- Maximum Likelihood and Maximum Apriori estimates
- Overfitting and Regularization

What we have learned so far (Contd. . .)

- Distance based classification
- Bayes decision theory and Bayes classifier
- Supervised learning and some foundations
- Linear Regression and Logistic Regression
- Maximum Likelihood and Maximum Apriori estimates
- Overfitting and Regularization

What we have learned so far (Contd. . .)

- Distance based classification
- Bayes decision theory and Bayes classifier
- Supervised learning and some foundations
- Linear Regression and Logistic Regression
- Maximum Likelihood and Maximum Apriori estimates
- Overfitting and Regularization

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd. . .)

- Gradient Descent Algorithms
- Logistic and Linear Regression using Python
- Hyperplane based Classifiers and Peceptron
- Support Vector Machines
- Kernel Methods
- Feedforward Neural Networks
- Backpropagation Algorithm
- Different aspects of training neural networks
- Convolutional Neural Networks
- Recurrent Neural Networks

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd. . .)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

What we have learned so far (Contd...)

- Unsupervised Learning
- K-means clustering
- PCA
- Spectral Clustering
- Markov Random Fields
- MCMC methods
- RBMs
- Latent Variabel Models and GMMs
- Free Energy and EM algorithm
- Model Selection
- Making ML algorithms work: Bias and Variance

ML Zoo

Model Selection

How to choose the right model?

- Bias-Variance tradeoff
- Model simplicity
- Bayesian Information Criteria
- Feature Selection

How to debug a machine learning model?

- Detecting bias vs variance
- Choosing the correct hyperparameters
- Ablation studies
- Optimization Issues

Choosing the Right Model

- **Bias-Variance tradeoff**
  - Too few parameters - underfitting - bad performance on both test and training set
  - Too many parameters - overfitting - good performance on training set, bad performance on test set

- **Model simplicity**
  - Always prefer a simpler model - Occam's Razor
  - Simpler models are easy to interpret and debug
  - Simpler models also offer computational advantage

Choosing the Right Model (contd. . .)

- **Bayesian Information Criteria**
  - $BIC(\theta) = k \log n - 2 \log \mathcal{L}(\mathcal{D}; \theta)$
  - k = Number of parameters, n = sample size, $\mathcal{L}(\mathcal{D}; \theta)$ = likelihood of observed data evaluated at θ
  - Choose the model with minimum BIC

- **Feature Selection**
  - Unnecessary to keep features that are: (a) highly correlated with each other or (b) not correlated with target
  - PCA for solving (a), correlation analysis for solving (b)

Detecting Bias vs Variance

- Plot the learning curve
- Bias - Both training and test errors are unacceptably high
- Variance - Training error decreases quickly but test error is high
- Use cross-validation to find the right model

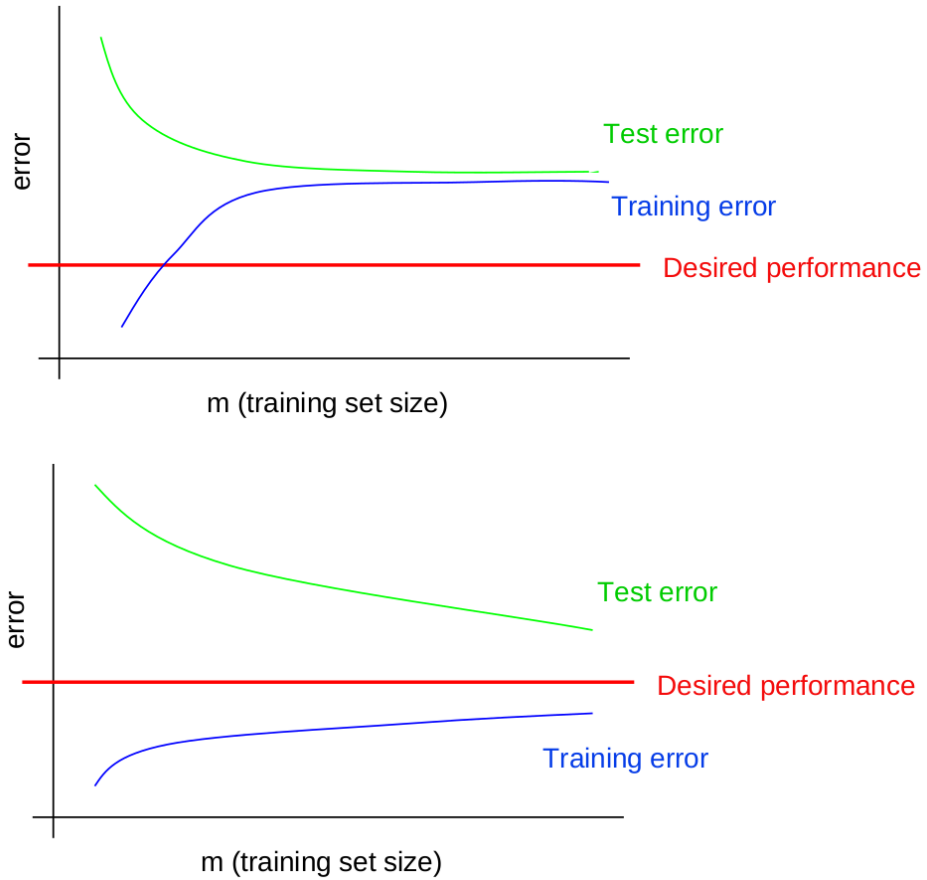


Figure 2: Bias (top) vs Variance (bottom

Debugging Machine Learning Models

- **Choosing correct hyperparameters**
  - Use cross validation
- **Ablation Studies**
  - Check which parts in the pipeline are not working by replacing them with oracles
  - Always find bottlenecks before trying to make changes
- **Optimization Issues**
  - Is the cost function meaningful?
  - Has the optimization procedure converged?
  - Are the gradients too noisy?

Choosing the right algorithm

- Is the problem supervised/unsupervised?
- Is it a classification problem? Regression? Clustering? etc.
- How much data is available?
- How much computational power is available?
- What is the desired level of accuracy?
- Is model interpretability a requirement?
- Start by exploring the data - for example by plotting it
- Start with the simplest strategy and optimize performance bottlenecks at each iteration
- https://docs.microsoft.com/en-us/azure/machine-learning/studio/algorithm-choice offers a practical summary

Glossary

Concept	Comments
Activation Function	Non-linearity inducing function used in neural networks. Eg. tanh, sigmoid, ReLU etc.
Ancestral Sampling	Sampling technique for directed graphical models
Autoregressive Model	Models in which value at time t is a function of values at time $1, \ldots, t-1$
Backpropagation	The algorithm used for computing gradients in a neural network
Bayesian Information Criteria	A criteria used for selecting a model out of a finite number of models
Bayesian Network	A directed, acyclic graph that encodes a joint probability distribution over the variables on its nodes by using condi-

Glossary (contd. . .)

Concept	Comments
Belief Propagation	An algorithm used for performing inference in probabilistic graphical model
Bernoulli Random Variable	Random variable that takes two values 0 and 1
Beta Random Variable	Random variable that takes values in the range $[0, 1]$. Usually used as a conjugate prior for Bernoulli
Binomial Random Variable	Random variable that takes non-negative integer values up to a pre-specified integer n
Canonical Correlation Analysis	A method used for analyzing the relationship between two random vectors
Classification	A supervised learning problem where the target variable takes its values in

Glossary (contd. . .)

Concept	Comments
Clustering	An unsupervised learning problem where the input data has to be partitioned into meaningful groups
Convolutional Neural Network	A neural network that exploits spatial regularity in input data. Usually used when input is in the form of images
Cost Function	The optimization objective that is solved by a learning problem. Also known as a loss function
Cross-Validation	A method of performing validation for tuning model hyperparameters
Data Augmentation	Extending the training set by artificially injecting variations of existing data to create new examples

Glossary (contd. . .)

Concept	Comments
Decision Tree	A method commonly used for classification
Density Estimation	The problem of estimating the probability distribution from which training examples have been drawn
Dirichlet Distribution	A probability distribution over random vectors whose entries are non-negative and add up to one
Discriminant Function	A function that is used to predict the class to which an input example belongs in a classification setting
EM Algorithm	Expectation maximization algorithm. An algorithm for performing maximum likelihood estimation in models with la-

Glossary (contd. . .)

Concept	Comments
Exponential Distribution	A distribution over positive real valued random variable with a specific exponential form for pdf
Factor Analysis	Probabilistic method for finding a lower dimensional subspace in which the observed data resides
Feature Extraction	Extracting meaningful features from raw data that are used by a machine learning algorithm
Forward-Backward Algorithm	The algorithm used in training Hidden Markov Models
Gaussian Distribution	Also known as normal distribution. One of the most heavily used distributions

Glossary (contd. . .)

Concept	Comments
Gaussian Process	A stochastic process where every finite subset of random variables is jointly Gaussian. Used for classification and regression. Also offers confidence estimates
Generalization	The ability of a machine learning model to work well on previously unseen data
Generative Model	A probabilistic model from which observations of interest can be sampled to mimic the training set
Gradient Descent	An optimization algorithm based on first order derivative of a function
Graphical Model	A graph representing a probability distribution over a set of random variables

Glossary (contd...)

Concept	Comments
Hidden Markov Models	Models commonly used for sequential/-time series data
Hidden/Latent Variable	An unobserved random variable in a model
Independent Component Analysis	A dimensionality reduction procedure like PCA
Importance Sampling	A sampling strategy
IID	Independent and identically distribution
Inference	The task of finding a distribution over unobserved random variables conditioned on the observed random variables

Glossary (contd. . .)

Concept	Comments
K-Means	A clustering algorithm
Kernel Function	A positive definite function that measures similarity between its two inputs
KL-Divergence	A measure of distance between two probability distribution. It is not a mathematical distance
Kriging	Regression using Gaussian process
Laplace Approximation	Approximating a probability distribution locally using a Gaussian based on the second order derivative
Lasso	A variant of linear regression with regularization
Linear Regression	A method used for regression where the regression function is assumed to be lin-

Glossary (contd. . .)

Concept	Comments
Logistic Regression	A method used for two class classification problems
LSTM	A type of recurrent neural network used for sequential data
MAP	Maximum aposterior - the maximizer of the posterior distribution over a set of unobserved random variables
Markov Chain	A stochastic process where the distribution of a random variable at time t depends only on the random variable at time $t-1$
Markov Chain Monte Carlo	A method for sampling from a joint probability distribution over several variables

Glossary (contd. . .)

Concept	Comments
Mixture Model	A probabilistic model in which the observed variable is assumed to be drawn from a mixture of underlying latent distributions
Multilayer Perceptron	A fully connected, feed forward neural network
Naive Bayes Model	A model used for classification that relies on conditional independence of all features when the class label is observed
Online Learning	Learning in a setting where examples arrive sequentially
Over-fitting	The situation when a model performs good on training set but poorly on test

Glossary (contd...)

Concept	Comments
Perceptron	Like a single layer neural network with a step function as the activation function
Posterior	The probability distribution over unobserved random variables given observed random variables
Regression	A learning problem where the target variable that is to be learned is continuous
Regularization	Any method aimed at improving the generalization performance of a learning algorithm
Rejection sampling	A sampling method
Ridge Regression	A form of linear regression with regu-

Glossary (contd. . .)

Concept	Comments
Simplex	A set of vectors whose entries are non-negative and sum up to one
Singular Value Decomposition	A decomposition method for matrices that expresses a matrix $\mathbf{X}$ as $\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^\intercal$
Skip Connections	Connections in a neural network between non-consecutive layers
Softmax Function	A function that transforms a vector of real numbers into a vector of same dimension from a simplex
Steepest Descent	Gradient descent using the direction of negative gradient at each step
Support Vector Machines	A method used for classification with large margin

Glossary (contd. . .)

Concept	Comments
Training Set	A set of examples that are used to train the machine learning model
Uniform Distribution	A distribution over an interval $[a, b]$ where each point in the interval is equally likely
Validation Set	A set of examples that are used for cross-validation to tune model hyper-parameters
Variational Inference	An approximate method for performing inference in complicated probabilistic models